Die Hausaufgaben müssen von jedem Studierenden einzeln bearbeitet und abgegeben werden. Für die Hausaufgabe sind die aktuellen Informationen vom Blog (https://seblog.cs.uni-kassel.de/ws1819/programming-methodologies/) und aus den Übungen zu berücksichtigen.

Die Abgabefrist läuft jeweils bis Beginn der nächsten Übung.

Abgaben per Mail werden nicht akzeptiert.

Legen Sie sich ein neues Githubprojekt mit folgendem Link an!

https://classroom.github.com/a/LDet7ZoT

Hinweis:
Diese Hausaufgabe fasst den gesamten Inhalt der Veranstaltung in einem kleinen Spiel zusammen. Es wird ein objektorientiertes Vorgehen vorausgesetzt, d.h. konkrete Gegenstände und Dinge werden als Objekte, Beziehungen als Links abgebildet. Nur Eigenschaften sind als Attribute zu repräsentieren.

Vorbereitung:
Es ist das Package ''de.uks.se.tictactoe" zu verwenden. Angaben zu Benennungen oder Speicherorten in spitzen Klammern (z.B. <Projekt Ordner>) sind durch die jeweils zutreffenden Texte zu ersetzen.

TicTacToe Rules:
TicTacToe is a 3x3 square game. Two players alternately fill an empty square with their icon (X or O). The player with 3 icons in one row, column or diagonal line wins the game.

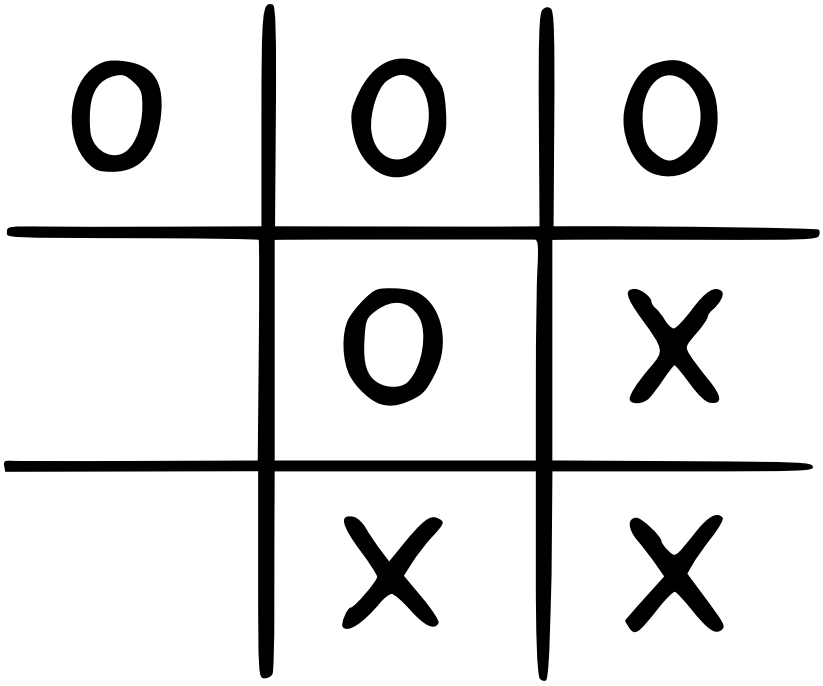

Abbildung 1: TicTacToe

Aufgabe 1: Szenarien (20P)

Erstellen Sie zwei unterschiedliche textuelle, objektorientierte Szenarien in englischer Sprache, welche sich an den folgenden Beschreibungen orientieren:

Aufgabe 1.1: Alice platziert ein Symbol

Alice und Bob spielen gegeneinander und Bob hat bereits ein Symbol platziert. Nun platziert Alice ein Symbol.
Das Szenario ist unter <Projekt Ordner>/doc/scenario1.txt abzulegen.

Aufgabe 1.2: Bob gewinnt die Partie

Alice und Bob spielen gegeneinander und Bob platziert sein Symbol so, dass er beispielsweise eine Reihe, Spalte oder Diagonale füllt.
Das Szenario ist unter <Projekt Ordner>/doc/scenario2.txt abzulegen.

Aufgabe 2: Objektdiagramme (20P)

Leiten Sie aus beiden textuellen Szenarien aus Aufgabe 1 jeweils ein Objekdiagramm zu Start- und Endsituation ab. Es sind also **vier** einzelne Objektdiagramme zu erstellen. Verwenden Sie hier ein Grafik- oder Diagrammwerkzeug ihrer Wahl (kein Codegenerierungswerkzeug!) oder zeichnen Sie das Diagramm händisch. Die Diagramme sind unter <Projekt Ordner>/doc/<start|end><Szenarionummer>.png abzulegen.

Aufgabe 3: Klassendiagramm und Fulib Modell (20P)

Leiten Sie ein Klassendiagramm ab und implementieren Sie dieses als Fulib Klassenmodell:

Aufgabe 3.1: Klassendiagramm

Aus den in Aufgabe 2 erstellten Objektdiagrammen ist **ein** Klassendiagramm abzuleiten. Verwenden Sie hier ebenfalls ein Grafik- oder Diagrammwerkzeug ihrer Wahl (kein Codegenerierungswerkzeug!) oder zeichnen Sie das Diagramm händisch. Das Diagramm ist unter <Projekt Ordner>/doc/classDiag.png abzulegen.

Aufgabe 3.2: Fulib Modell

Erstellen sie eine Java-Datei ''TicTacToeClassModel.java" im Package ''de.uks.se.tictactoe", in welcher Sie einen JUnit Test anlegen. In der Testmethode ist das Klassendiagramm als

Fulib Model Code zu implementieren. Generieren Sie abschließend Java Quellcode in den Ordner ''src" ins Package ''de.uks.se.tictactoe.model".

Aufgabe 4: Storyboards und JUnit (20P)

Erweitern Sie zunächst ihr Projekt, indem sie folgende Klassen hinzufügen. Die Klasse, welche das Spiel (z.B. TicTacToeGameController) kontrollieren soll einen leeren Methodenrumpf ''public boolean checkWinner()". Und einen leeren Methodenrumpf ''public boolean placeIcon(..)". Der Typ des Parameters muss der Klasse entsprechen, welche ein Symbol repräsentiert, z.B. Icon icon.

Erstellen Sie anhand der Szenarien aus Aufgabe 1 zwei Storyboards als separate JUnit Test Klassen im Package ''de.uks.se.tictactoe.tests":

- ''AlicePlacesAnIcon.java" - Es ist über Asserts zu prüfen, ob das Platzieren korrekt ausgeführt wurde. Als Aktion dient der Aufruf der Methode placeIcon(..).
- ''BobWins.java" - Es ist über Asserts zu prüfen, ob Bob gewonnen hat. Als Aktion dient der Aufruf der Methode placeIcon(..), welche ihrerseits zur Vereinfachung intern am Ende selbstständig die Methode checkWinner() auf dem Spiel-Objekt aufruft.

Aufgabe 5: Logik(20P)

Implementieren Sie die Rümpfe der in Aufgabe 4 erstellten Methoden placeIcon(..) und checkWinner(), sinnvoll und unter Beachtung der Regeln, bis die in Aufgabe 4 erstellten Storyboards erfolgreich durchlaufen. Zur Vereinfachung muss die Methode placeIcon(..) intern am Ende selbstständig die Methode checkWinner() auf dem Spiel-Objekt aufrufen. Hier ist Java Code zu verwenden.